REMARKS

Claims 1-34 are pending in the application of which 22-34 are added by this Amendment. As required by 37 CFR § 1.121, Applicant submits a version with markings showing changes to the application. In light of the amendments and following remarks, Applicant believes all the pending claims are now in condition for allowance.

Claims 4-5, 7-8 and 10 were objected to be indicated would be allowed if rewritten in independent form. Applicant has amended these claims accordingly so they are now allowable. Additionally, new claims 24-25, 27-28 and 30 have similar features so it is believed they are allowable for at least the same reasons.

The § 102(b) Rejection of Claims 1-3, 6 and 11-15

Claims 1-3, 6 and 11-15 were rejected under 35 USC § 102(b) as allegedly being anticipated by European Applicant No. 869433 by Anodide et al. (hereinafter "Anodide"). Therefore, the Office Action is alleging that Anodide teaches all the features of the claims. For the following reasons, Applicant respectfully traverses the rejection.

Anodide describes that in order to inform the test software about the windows of an application, the tester manually opens and closes each window. For example, Anodide states as follows:

> The tester "informs" TDE what the windows of the application are, by sequentially opening and closing each one.

(col. 8, lines 6-8). After being informed of the windows in this manner, Anodide discusses creating an object forest and opens relation (see, e.g., col. 8, lines 48-56). Thus, Anodide discusses that the tester manually opens and closes windows to inform the test software about the GUI windows.

The Office Action has not shown where Anodide teaches identifying a first set of windows, identifying a second set of windows and comparing the sets of windows to identify a new window as claimed. For example, claim 1 recites as follows:

> identifying a first set of windows that are active on the desktop of the computer;
>
> identifying a second set of windows that are active on the desktop of the computer;

comparing the first set of windows to the second set of windows to identify a new window in the second set;

The Office Action attempts to show that Anodide describes these features, but the following will show that a closer examination of Anodide reveals this is not the case.

The Office Action stated on page 3 that creating the GUI object forest and (op, win) pairs is equivalent to identifying a first set of windows that are active on the desktop of the computer. The GUI object forest and (op, win) pairs in Anodide refer to the GUI of the application under test as a whole, meaning that the GUI object forest and (op, win) pairs do not in any way relate to what windows are currently active on the desktop (see, e.g., col. 8, lines 48 to col. 9, line 25). Anodide therefore has not been shown to identify a first set of windows that are active on the desktop as claimed.

The Office Action then stated on page 4 that the same GUI object forest and (op, win) pairs anticipates the second set of windows that are active on the desktop of the computer. Applicant does not believe that the GUI object forest and (op, win) pairs can be BOTH the first and second set of windows that are active on the desktop as claimed. Even assuming this is true for the sake of argument, the Office Action stated later on page 5 that it is implied in Anodide that the first and second set of windows are compared as claimed.

Anodide has not been shown to teach comparing two sets of windows as claimed to identify a new window. In fact, Anodide teaches away from the invention claimed as the reference has been cited to show a user that manually opens and closes windows in an application under test to identify windows. That is not what is claimed and therefore a prima facie of anticipation has not been established so the claims are patentably distinct over the reference.

The § 103(a) Rejection of Claim 9

The Office Action rejected claim 9 under 35 USC § 103(a) as allegedly being unpatentable over Anodide in view of U.S. Patent No. 5,600,789, issued February 4, 1997 to Parker et al. (hereinafter "Parker"). For the following reasons, Applicant respectfully traverses the rejection.

Initially, claim 9 is a dependent claim and the Office Action has not shown where the additional reference remedies the deficiencies of the primary reference Anodide as shown above. Accordingly, claim 9 is patentably distinct for at least the same reasons as above.

Moreover, the Office Action has not shown that the references, even if combined, disclose receiving input from a user that one or more graphical interface objects should be ignored when generating a map as claimed. The Office Action states that Parker discloses that when comparing windows, a user can mask out areas of a window (citing col. 29, line 5).

Assuming this is true for the sake of argument, Anodide discusses a user manually opening and closing windows to identify new windows. Applicant does not see how it is suggested that a user inform a program to mask out areas of window when the user is the one that is accessing those objects in the windows to open and close the windows. Accordingly, even the combination of the cited references does not disclose or suggest the invention recited in claim 9 so the claim is patentably distinct over the references.

The § 103(a) Rejection of Claim 16

The Office Action rejected claim 16 under 35 USC § 103(a) as allegedly being unpatentable over Anodide in view of the state of the art. Claim 16 is a dependent claim and is patentably distinct for at least the same reasons as above with regard to the independent claim.

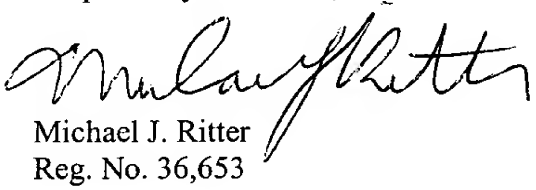
The § 103(a) Rejection of Claims 17-21

The Office Action rejected claims 17-21 under 35 USC § 103(a) as allegedly being unpatentable over Anodide in view of U.S. Patent No. 6,189,116, issued February 13, 2001 to Mongan et al. (hereinafter "Mongan"). In a sincere effort to expedite prosecution, Applicant amended claim 17 to include features similar to those in claim 1. Applicant reserves all right to pursue the original or other claims in a continuing application. Mongan is cited as allegedly disclosing a test generator, but it has not been shown that Mongan remedies the deficiencies of the primary reference Anodide as discussed above in reference to claim 1. Accordingly, claims 17-21 are patentably distinct for at least the same reasons as above.

Conclusion

For the foregoing reasons, Applicant believes all the pending claims are in condition for allowance and should be passed to issue. If the Examiner feels that a telephone conference would in any way expedite the prosecution of the application, please do not hesitate to call the undersigned at (408) 446-8693.

Respectfully submitted,



Michael J. Ritter
Reg. No. 36,653

RITTER, LANG & KAPLAN LLP
12930 Saratoga Ave., Suite D1
Saratoga, CA 95070
Tel: 408-446-8690
Fax: 408-446-8691

VERSION WITH MARKINGS TO SHOW CHANGES MADE TO THE APPLICATION

In the Specification

The paragraph that begins on page 1, line 13 has been amended as follows:

This application is related to U.S. Patent Application No. 08/655,149, filed May 30, 1996, now issued as U.S. Patent No. 5,754,760, which is hereby incorporated by reference.

In the Claims

Claims 4, 7-8, 10, and 17 have been amended as follows:

4. (Amended) A computer implemented method of mapping a graphical user interface of an application, comprising: **[The method of claim 2,]**

identifying a first set of windows that are active on the desktop of the computer;

performing an action on a graphical user interface object in a window of the application;

identifying a second set of windows that are active on the desktop of the computer;

comparing the first set of windows to the second set of windows to identify a new window in the second set;

analyzing the map to determine if the new window is already present in the map, wherein the new window is determined to already be present in the map if similarities between the new window and the window in the map are above a similarity threshold; and

adding the new window to a map of the graphical user interface of the application.

7. (Amended) A computer implemented method of mapping a graphical user interface of an application, comprising: **[The method of claim 2, further comprising]**

identifying a first set of windows that are active on the desktop of the computer;

performing an action on a graphical user interface object in a window of the application;

identifying a second set of windows that are active on the desktop of the computer;

comparing the first set of windows to the second set of windows to identify a new window in the second set;

analyzing the map to determine if the new window is already present in the map;

adding the new window to a map of the graphical user interface of the application; and

receiving input from a user that two or more windows of the map that have been determined as different should be considered the same window.

8. (Amended) A computer implemented method of mapping a graphical user interface of an application, comprising: **[The method of claim 2, further comprising]**

identifying a first set of windows that are active on the desktop of the computer;

performing an action on a graphical user interface object in a window of the application;

identifying a second set of windows that are active on the desktop of the computer;

comparing the first set of windows to the second set of windows to identify a new window in the second set;

analyzing the map to determine if the new window is already present in the map;

adding the new window to a map of the graphical user interface of the application; and

receiving input from a user that two or more windows of the map that have been determined as the same should be considered different windows.

10. (Amended) A computer implemented method of mapping a graphical user interface of an application, comprising: **[The method of claim 1, further comprising]**

identifying a first set of windows that are active on the desktop of the computer;

performing an action on a graphical user interface object in a window of the application;

receiving input from a user specifying an amount of time to wait after performing the action before identifying a **[the]** second set of windows;

identifying the second set of windows that are active on the desktop of the computer;

comparing the first set of windows to the second set of windows to identify a new window in the second set; and

adding the new window to a map of the graphical user interface of the application.

17. (Amended) A system for testing applications, comprising:

an application mapper that programmatically executes an application to generate a map of the graphical user interface of the application, the application mapper adding a new window to the map by performing an action in the graphical user interface and identifying the new window by comparing windows in the graphical user interface before and after the action;

a script generator that utilizes the map to generate scripts that include instructions to test the application; and

an application tester that executes the scripts to test the application.